**Frontend Development with React.js**

**Project Documentation format**

1. **Introduction**
   * **Project Title**:The project title is "CookBook", an advanced React.js application for culinary enthusiasts.
   * **Team Leader:** RAMYA.R ([ramyaramya2723@gmail.com](mailto:ramyaramya2723@gmail.com))

* **Team members**: PRINCE MARIYA.A ([princemariya24.06.2002@gmail.com](mailto:princemariya24.06.2002@gmail.com))

ASHVATHI.E ([ashvathielango0905@gmail.com](mailto:ashvathielango0905@gmail.com))

DHANALAKSHMI.G ([g.dhanalakshmi003@gmail.com](mailto:g.dhanalakshmi003@gmail.com))

1. **Project Overview**
   * **Purpose**: The purpose is to provide a recipe exploration platform, with key features including recipe browsing with lazy loading, category navigation, search functionality, detailed recipe views with YouTube videos, and a responsive "Village Cooking Theme" design. The scenario involves a user, Sarah, using the app to find and learn Indian recipes, highlighting its educational potential.
   * **Features**: The CookBook is a React.js application designed for culinary enthusiasts to explore and learn recipes. It features recipe browsing with lazy loading, category navigation, search functionality, and detailed views with YouTube video tutorials, all styled with a "Village Cooking Theme" for a warm, earthy aesthetic.
2. **Architecture**
   * **Component Structure**: The application is built with React.js, using react-router-dom for navigation, Axios for API calls, and integrating with <https://recipe_backend_tgsm.onrender.com/recipes> (noted as inactive) and YouTube API. State management is handled locally with React's built-in hooks (useState, useEffect), with no mention of global state libraries like Redux. The design is responsive across devices, with performance monitoring and testing setups included.
   * **State Management**: Local state with React's useState and useEffect; no global state management library.Architecture Notes: Responsive design with performance monitoring tools, noting the API (<https://recipe_backend_tgsm.onrender.com/recipes>) appears inactive, suggesting a hypothetical setup.
   * **Routing**: Frontend: React.js
     + - Routing: react-router-dom for navigation
       - Data Fetching: Axios for API calls, integrating with <https://recipe_backend_tgsm.onrender.com/recipes> and YouTube API.
3. **Setup Instructions**
   * **Prerequisites**:
     + - Node.js installed
       - Basic knowledge of HTML, CSS, JavaScript, and React.js
       - Git installed
       - Visual Studio Code or preferred text editor
       - YouTube API key for video integration
   * **Installation**:
     + - Clone the Git repository: git clone https://github.com/yourusername/cookbook-dashboard
       - Navigate to the project directory: cd cookbook-dashboard
       - Install dependencies: npm install
       - Start the development server: npm start
       - Access the application at <http://localhost:3000>
4. **Folder Structure**
   * **Client**: The structure includes a public folder for assets (images, favicon), and src/ with components (Navbar, Hero, RecipeList, RecipeCard, Footer), features/recipes (Home, Category, SearchResults, RecipeDetails), hooks (useRecipes), and root-level files like App.js, index.js, App.css, and index.css.
   * **Utilities**:
   * **public/:** Contains assets like images and favicon
   * **src/:**
     + **components/:** Navbar, Hero, RecipeList, RecipeCard, Footer
     + **features/recipes/:** Home, Category, SearchResults, RecipeDetails
     + **hooks/:** useRecipes
     + **App.js:** Root component
     + **index.js:** Entry point of the application
     + **App.css and index.css:** Styling files
5. **Running the Application**

The command to start is npm start, accessing the application at <http://localhost:3000>.

* + **Frontend**: Command: npm start.Access the application at <http://localhost:3000>.

**7.Component Documentation**

* + **Key Components**: Major components include Navbar (navigation and search), Hero (welcome section), RecipeList (list of recipes), RecipeCard (individual previews), Footer (copyright info), Home (landing page), Category (cuisine filters), SearchResults (search queries), RecipeDetails (detailed views with videos), and useRecipes (data fetching hook). Each has specific purposes and props, detailed as follows:
    - * Navbar.jsx: Navigation and search, props: none, includes links and responsive design.
      * Hero.jsx: Welcoming section, props: onGetStarted, features gradient background.
      * RecipeList.jsx: Renders recipe cards, props: recipes, maps over recipe array.
      * RecipeCard.jsx: Individual previews, props: recipe, shows image and details.
      * Footer.jsx: Copyright info, props: none, styled with theme.
      * Home.jsx: Landing page, props: none, includes lazy loading.
      * Category.jsx: Cuisine filters, props: category, uses URL parameters.
      * SearchResults.jsx: Search results, props: searchTerm, filters by query.
      * RecipeDetails.jsx: Detailed views, props: recipeId, embeds YouTube videos.
      * useRecipes.js: Data fetching, returns recipes, isLoading, error, uses Axios.
  + **Reusable Components**: See the detailed list above under Component Documentation, with purposes and props for each.

**8.State Management**

* + **Global State**: State is managed locally with React's useState and useEffect, with the useRecipes hook handling recipe data fetching. No global state management library is mentioned, suggesting each component manages its own state.
  + **Local State**: The application uses React's built-in hooks (useState, useEffect) for local state management.No global state management library is used; each component manages its own state, with useRecipes handling recipe data.

**9.User Interface**

The UI follows a "Village Cooking Theme" with custom CSS variables for a warm, earthy aesthetic, including components like Navbar, Hero, RecipeList, and more. It's designed to be responsive across devices, with performance monitoring.

 **Final Entry:**

* **Design:** "Village Cooking Theme" with custom CSS variables for a warm, earthy aesthetic
* **Components:** Navbar, Hero, RecipeList, RecipeCard, Footer, etc.
* **Responsive Design:** Optimized for various devices with CSS media queries

**10 .Styling**

* **CSS Frameworks/Libraries**: Styled with custom CSS variables for colors and layouts, with responsive design breakpoints, aligning with the "Village Cooking Theme".
* **Theming**:
* Uses custom CSS variables for consistent theming across the application
* Responsive design with breakpoints to handle different screen sizes

1. **Testing**

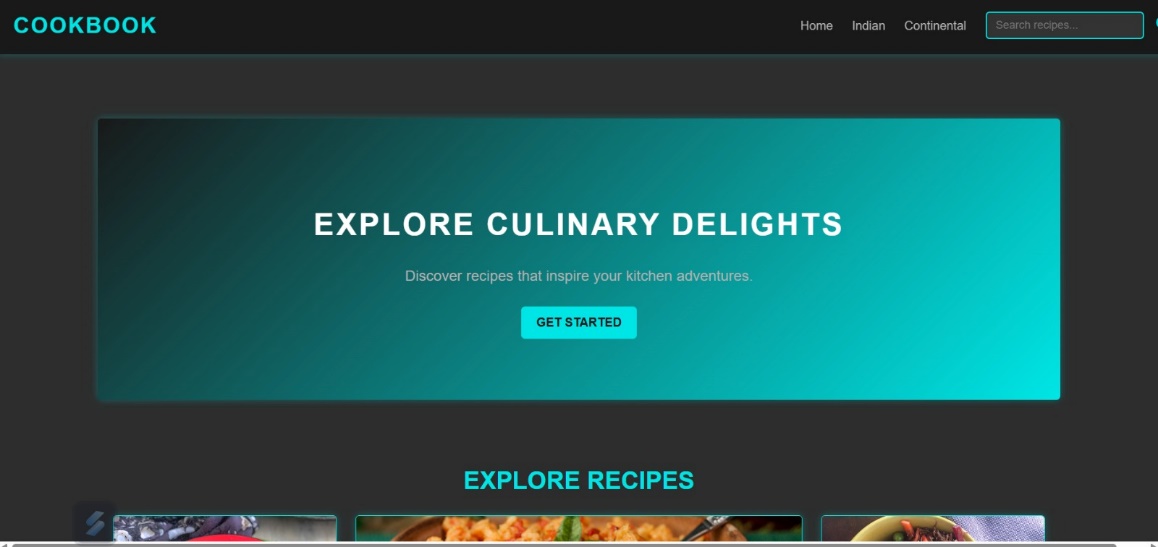
* **Testing Strategy**: Testing is implemented with Jest for unit tests, and performance monitoring is handled with reportWebVitals.js, ensuring code coverage and optimal user experience.
* **Code Coverage**:

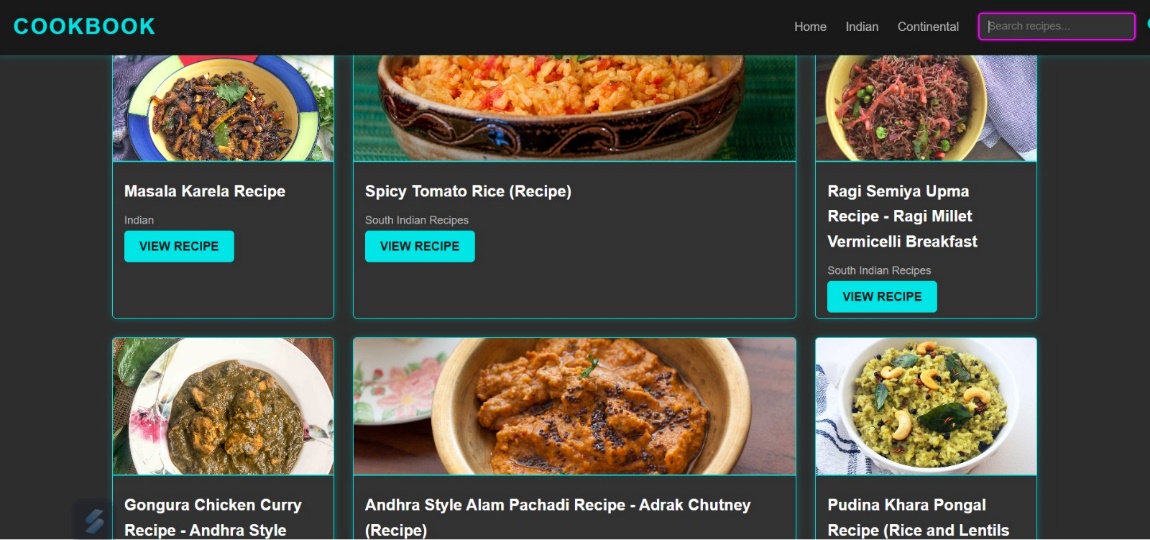
 **Unit Testing:** Implemented using Jest

 **Performance Monitoring:** Uses reportWebVitals.js to monitor web performance metrics

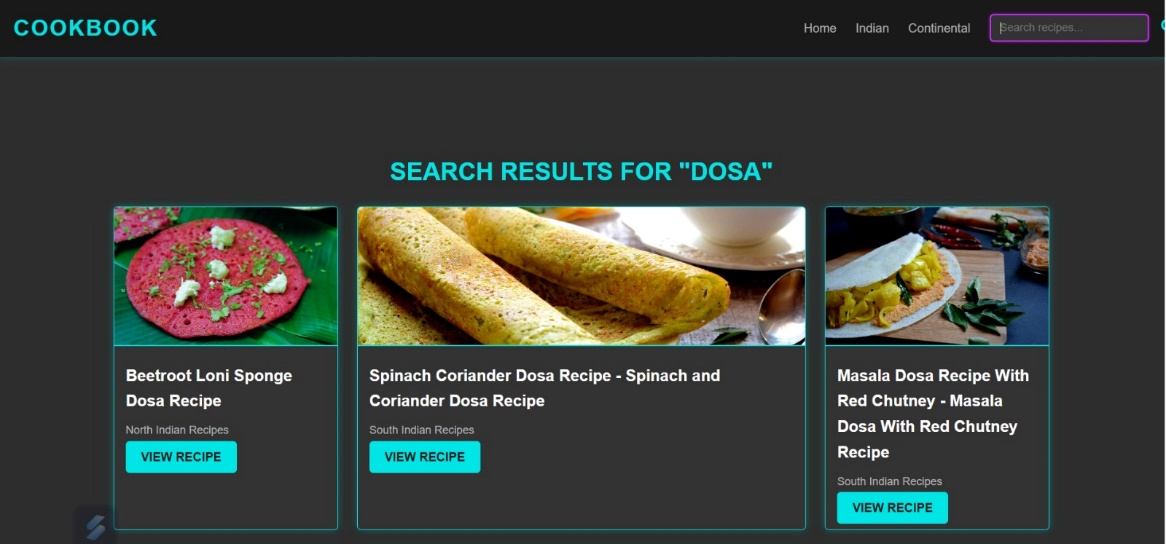
1. **Screenshots or Demo**

**1.Home Page:** Features the Hero section with a gradient background, a grid of RecipeCards with "Load More," and a Footer with a warm tone.

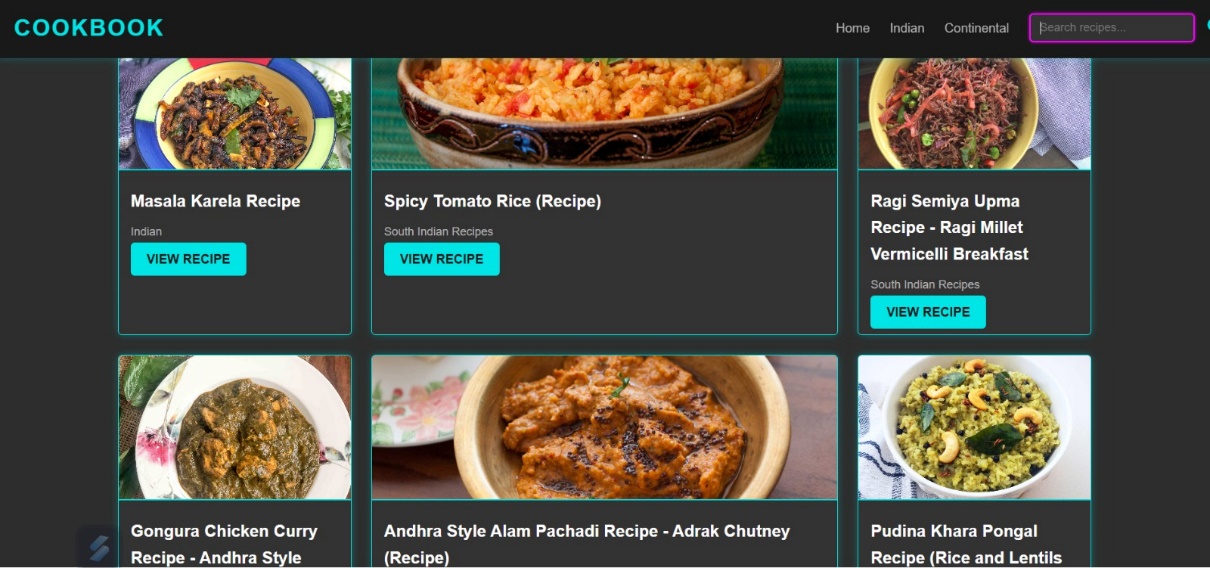




**2.Search Results Page:** Shows filtered recipes with a centered title.

****

* 1. **Category Page**: Displays a filtered RecipeList in a responsive grid.



Demo: <https://drive.google.com/file/d/1Gb79gZpDfqCUXIhUMWUtktI2w5-3x1Wi/view?usp=drive_link>

1. **Known Issues**

 **Content from CookBook:** No known issues are specified, suggesting the application is robust and scalable.

1. **Future Enhancements**

 Suggests potential enhancements like implementing user accounts for personalized recipe collections and adding offline support for recipe viewing.